to be presented in 1995 NASA workshop on Science Information and Data Compression

A Comparison of Model-Based VQ Compression with other VQ Approaches

Mareboyana Manohar
Department of Computer Science
Bowie State University
Bowie, MD 20715

James C. Tilton
Information Science and Technology Branch
NASA Goddard Space Flight Center
Greenbelt, MD 20771

ABSTRACT

In our pervious work on Model-Based Vector Quantization (MVQ)[1], we presented some performance comparisons (both rate distortion and decompression time) with VQ and JPEG/DCT. In this paper, we compare the MVQ's rate distortion performance with Mean Removed Vector Quantization (MRVQ) and include our previous comparison with VQ. MVQ is similar to MRVQ in many ways. Both of these techniques extract means of the vectors (raster-scanned image blocks) and reduce them to mean removed residuals by subtracting block means from the elements of the vectors. In the case of MRVQ, a codebook of residual vectors is generated using a training set. For every vector from the input image, the block mean and address of the codevector from the codebook that matches the input vector closest are transmitted to the decoder. The codebook is generated using generalized Lloyd algorithm on training set of residual vectors. For MVQ the pairs consist of vector means and address of the closest matching vector from codebook generated by models based on statistical properties of the residuals and Human Visual System (HVS). In our experiments, we found that MVQ performance in rate distortion sense is almost always better than VQ and is comparable to MRVQ. Further, MVQ is much easier to use than either VQ or MRVQ, since the training and managing of explicit codebooks is not required.

1. INTRODUCTION

Vector Quantization (VQ) compression approaches are characterized by an asymmetric computational property which has been found to be very useful for image data archival, retrieval and distribution [2,8]. While VQ compression can be computationally demanding, decompression of the VQ coded data is a computationally light table lookup process. This means less computational burden on the user compared to some compression techniques that are characterized by symmetric computational requirements such as JPEG/DCT [3].

The performance of VQ compression approaches depend heavily on the quality of codebooks employed. Codebook generation involves a computationally intensive training process. One of the popular training algorithms for VQ codebook generation is the Generalized Lloyd Algorithm (GLA) [4]. See [5,6] for other training algorithms. The training process is carried out on a set of image data called the training data set. The codebook thus generated provides optimal performance (in the rate distortion sense) for a given size of the codebook for all the images included in the training set. However, for images outside this training set the performance is suboptimal. Nevertheless, by carefully selecting the training set, an acceptable level of performance can be achieved.

Codebook size is an important issue in optimal performance of VQ compression approaches. The compression ratio (CR, defined as the ratio of number of bits in the input image to the number of bits in the compressed image) for VQ approaches decreases as the logarithm of the codebook size, and distortion

(normally measured as mean squared error between the input image and reconstructed image) decreases linearly with increasing codebook size [7]. Therefore, it is important to consider large codebooks for better rate-distortion performance for VQ approaches. Large codebooks, however, require large computational requirements for coding. We have used a parallel full search implementation of VQ to solve the computational problems arising from large codebooks [8]. However, since the codebooks involved are large, we cannot afford to include codebook in every image that we compress to decode. Thus, generation and maintenance of codebooks pose some significant problems for VQ approaches to be viable options for data archival, retrieval and distribution.

In MRVQ [7, pp. 435-441] approach, the codebooks are more generic compared to VQ's codebooks. MRVQ's rate distortion performance is better than that of VQ. In this technique, a block of pixels from the input source is vectorized ($\mathbf{X}_i$) in raster scan order and decomposed into mean and residue <mi, Ri>. These two components are quantized separately. In this paper we have compressed {mi }using a JPEG lossless compression and **Ri** is compressed using VQ. The codebook for quantizing **Ri** is obtained by training using Generalized Lloyd algorithm on a training set that does not include the test image. At the decoding end, the codebook used for compression are assumed to be available for decompression. In our experiments we have not used any scalar quantization on the mean values of the vectors, which may improve the rate distortion performance marginally. Instead, we have used JPEG lossless compression which exploits the image correlations optimally. Further, we have also compressed the indexes of the codevectors that match the input vectors from the source using a Ziv-Lempel compression technique[10].

Model-Based VQ (MVQ) [1] is a variant of VQ which eliminates storage and transmission of codebooks. MVQ internally generates codebook based on error models and HVS models that use standard deviation of residual vector elements from the source. As far as the user is concerned it is a codebookless VQ variant. In this approach, the image is decomposed into square blocks of k pixels. The block means are computed and transmitted as the first approximation of the data. The mean removed residuals or errors from the source are generated using mathematical models and used as codebook. A uniform random number generator is used to generate random numbers and shaped according to Laplacian distribution by using an inverse function for filtering. The generated error values are then normalized to -1.0 to 1.0. The error values thus obtained are independent and identically distributed (i.i.d.). Since we attempt to model the actual error residuals from the source which are not independent, we need to process these error values to impose the actual error characteristics. We proposed in our pervious work [1] a perceptual weighting scheme of the Laplacian random source that constituted the codebook. The mean removed errors from the source are quantized using the model generated codebook and for each mean removed block (considered as vector by arranging the k pixels in the raster scan mode), an address of the codevector from the codebook that matches the input vector closest is transmitted. A pair consisting of block mean and address of the codevector from the codebook approximate the input vector with compression that depends on the size of the block and codebook size. In the following sections, we describe briefly the model-based VQ, the perceptual weighting scheme and the performance comparison MVQ with VQ and MRVQ.

2. MODEL-BASED VQ ALGORITHM

The MVQ algorithm implicitly generates its codebook using mathematical models based one image specific parameter (λ) from the source which are transmitted along with the coded image data so that decoding can be performed by generating the same codebook at decoding end. The model simulates the mean removed vectors of the source. The means (block means) from the source, m_i of vector $\mathbf{X}_i$, are scalar random variables given by

$$m_i = \frac{1}{k}\sum_j x_{ij} \tag{1}$$

where x_{ij} is j^{th} element of the vector $\mathbf{X_i}$. For all the vectors, we compute the vector means and transmit them after lossless compression such as JPEG/DPCM or Ziv-Lempel algorithm[10]. The second phase of the algorithm is to send residual information in compressed form. The residual vector for i^{th} block composed of scalar random variables is given by

$$\mathbf{R_i} = \mathbf{X_i} - m_i \mathbf{U} \tag{2}$$

where $\mathbf{U}$ is a unit vector of same dimension as input vector $\mathbf{X_i}$. In MVQ, the residual vector, $\mathbf{R_i}$, is represented by an index I to the model generated codebook (CB) entries:

$$\text{MVQ: } \mathbf{R_i} \text{ --> } I_i \tag{3}$$

In our implementation, the codebook size (*nc*, number of entries in the codebook) is 16,384 resulting in 14 bit representation for the index, I. The compression ratio (CR) that can be obtained from this scheme is given by

$$\text{CR} = \frac{k}{b + \log_2(nc)} \tag{4}$$

At the decoding end,

$$\tilde{R}_i = CB(I_i) \tag{5}$$

where, $\mathbf{R_i}$ is an approximation of $\mathbf{R_i}$. The reconstructed vector at the decoding end is given by

$$\hat{X}_i = m_i + \tilde{R}_i \tag{6}$$

The distortion between input vector, $\mathbf{X_i}$ and the reconstructed vector, $\tilde{\mathbf{X}}_i$ is expressed in terms of mean squared error (MSE). The MSE for all the vectors drawn from the source image is given by

$$D = \sum_i \sum_j (x_{ij} - \tilde{x}_{ij})^2 \tag{7}$$

The important element of the MVQ is the simulation of residual vectors of the source image and building the codebook of 16,384 codevectors from this process. The number of codevectors 16,384 was selected because of the MasPar size which is a 128x128 array.

The input image is decomposed into square blocks of 4x4, 5x5, etc., depending on the targeted compression ratio. The vector size, k (r xc), for a targeted compression ratio can be computed from Eq. 4, where r is the number of rows and c is the number of columns of the block of pixels from the source image data. For each block of pixels, the block mean is computed, compressed using appropriate lossless compression, and transmitted. The residual vectors are computed by removing the mean from the vector

formed by raster scan of the image blocks. These residual vector elements are used to compute the mean (λ) of the Laplacian distribution into which the elements of the residual vectors are assumed to fit.

$$p(r) = \frac{1}{2\lambda} e^{-\frac{|x|}{\lambda}} \tag{8}$$

where λ is computed from the standard deviation of the distribution of the vector elements

$$\lambda = \frac{\sigma_e}{\sqrt{2}} \tag{9}$$

where σ_e is the standard deviation of the residual vector elements treated as i.i.d. random variables. After computing λ from the residual vectors of the image, it is used to generate Laplacian distribution using uniform random number generator. If u_i is the i^{th} uniform random number (value -1.0 to 1.0) generated by uniform random number generator, then the random number that forms a Laplacian distribution with mean equal to λ is given by

$$v_i = -\lambda \log_e (1-u_i) \tag{10}$$

A + or - sign is randomly given to v_i.

The number of v_i's generated depend on the codebook size and vector size. For a vector size of *k* and codebook size of nc, the number of v_i's is equal to the product of *k* and *nc*. The random numbers are grouped by *k* elements to form *nc* vectors that in turn form a codebook. These random numbers are independent and identically distributed, whereas the residual vector elements show considerable correlations among them which are manifest in nonzero off diagonal elements of covariance matrix. One solution to tuning the codebook to the actual source is to impose characteristics of the human visual system.

The HVS (Human Visual System) system has been incorporated in JPEG/DCT with excellent results. In JPEG/DCT, the DCT coefficients in are quantized based on their significance to human perception. Human visual weighted DCT coefficients have been used for progressive image transmission by Chitprasert and Rao [9] with visually more pleasing results than unweighted DCT coefficients. One way to impose HVS properties on codebook vectors generated by model is to DCT transform the vectors, weight the result with HVS DCT weight matrix, and inverse DCT the transform the result. The HVS DCT weight matrix we have used for model generated residual vectors is shown in Fig. 1. Note that we have modified the coefficient weighting matrix in [9] by making the DC term 0 so that it can be applied to residual vectors with 0 DC value.

0.000	1.000	0.702	0.381	0.186	0.085	0.037	0.016
1.000	0.455	0.308	0.171	0.084	0.039	0.017	0.007
0.702	0.308	0.212	0.124	0.064	0.031	0.014	0.063
0.381	0.171	0.124	0.077	0.042	0.021	0.010	0.004
0.185	0.084	0.064	0.042	0.025	0.013	0.007	0.003
0.084	0.039	0.031	0.021	0.013	0.007	0.004	0.002
0.037	0.017	0.014	0.010	0.006	0.004	0.002	0.001
0.016	0.007	0.006	0.004	0.003	0.002	0.001	0.0006

Figure 1: The Human Visual Systems weight function of DCT coefficients.

The model codebook using HVS is generated as follows. A set of (k*nc) uniformly distributed random numbers $\{u_i\}$ are generated and are nonlinearly mapped to $\{v_i\}$ according to Eq. 10 to fit into Laplacian distribution of specified mean λ which is computed from the input source as discussed. Next the $\{v_i\}$ are grouped to form set of nc k-element vectors $\{\mathbf{C}_i\}$. The vectors then undergo DCT transformation as given below.

$$\mathbf{C}_i^{dct} = \mathrm{DCT}(\mathbf{C}_i) \tag{12}$$

The next step is to weight each coefficient of and take the inverse DCT transform.

$$\mathbf{C}'_i = \mathrm{IDCT}(\mathbf{C}_i^{dct} * \mathbf{W}) \tag{13}$$

where * stands for element by element product of the two matrices. If $\mathbf{C}_i$ is smaller than 8x8 block (vector size less than 64), the corresponding subset of the W matrix is used in the product. The codebook containing $\{\mathbf{C}_i\}$ is now replaced by $\{\mathbf{C}'_i\}$ and the resulting codebook produces much better rate-distortion performance.

3. IMPLEMENTATION ON THE MASPAR

MVQ is an asymmetrical algorithm like all VQ approaches. While decoding is a table look-up process that can be performed quite efficiently on a sequential machine, coding using full search is computationally very intensive. Others solve VQ coding computational problems by structuring the codebook as tree [6]. However, the price of such structuring is suboptimal performance. We instead solve the computational problem by using an implementation on a massively parallel computer to obtain optimal performance (in the rate-distortion sense) for a given codebook size by doing full search on the codebook for best match of the source residual.

We have implemented MVQ coding on 16,384 processor MasPar MP-2. Each processing element (PE) has a local memory of 64 Kbytes. We generate 16,384 vectors of using the algorithm given in the preceding section and load each vector in each of the 16,384 PEs. We chose 16,384 vectors because we have found that this size codebook gives good performance, and because this size is most efficient on a 16,384 PE MasPar. The input image is decomposed into blocks of 4x4 or 6x6 or 8x8 pixels depending on the rate-distortion requirements and the block means are computed on the MasPar by propagating each vector from the image to one PE and the mean removed residuals from the input image are computed in parallel. The mean of the vectors are stored in a separate file and compressed using a JPEG's lossless compression [3] technique. The input residual vector elements are normalized with respect to maximum residual vector element value from the mean removed source. This can be accomplished by using

MasPar's min, max functions that take time proportional to the product of number of bits per pixel and clock cycle of the MasPar. Now, the normalized residual vectors are coded one by one by propagating each one to all PEs. The distance between the normalized residual vector from the input source and the codebook entries can be computed and the min distance can be found by using min function of the Maspar. The time taken for the best match search from the codebook takes time proportional to the number of bits in the Euclidean distance norm between source residual vector and codebook entry (32 bits).

If the codebook size is smaller than the PE array size by a factor of 2 or its multiple, simple load balancing techniques can be used to gain speedups for smaller codebooks. For example, if the codebook size is half the array size, then each codebook entry can be split up into two smaller codebook entries each containing half the number of elements and loaded into local memories of the adjacent PEs. The input residual vector from the source is similarly split into two smaller vectors and propagated such that first half of the vector is placed in PEs with even address and second half in those with odd address PEs. The distances are then computed in each PE separately and combined by shift operations to find the closest matching vector. Codebooks larger than PE array size by factor of 2 or its multiples can be handled by using processor virtualization.

In decoding, we retrieve the λ from coded file and generate the codebook using the λ and the DCT coefficient weighting function given in Figure 1. After the codebook is generated from λ, the rest of the process of decoding is a simple table lookup process. From the coded file, index for each residual vector is retrieved and the corresponding residual vector is read from the codebook. When the block mean is added to this residual vector, the block is reconstructed.

4. COMPARISON OF THE RESULTS WITH VQ AND MRVQ

The MRVQ requires that a codebook be generated through some training process. We used a set of Landsat TM images to generate four different codebooks for different vector sizes (k's). These codebooks can be used to compress a given TM image to the required compression ratio. We assume that all these codebooks are available to the decoder to reconstruct the images. A different set of codebooks was generated for different vector sizes for use with VQ. We used Generalized Lloyd Algorithm (GLA) to construct these codebooks. In MVQ, the codebook is generated using a model that needs single parameter (λ) derived from the image and the HVS weight matrix (which is not input image dependent). Thus, we need to send only one parameter (λ) to the decoder, so that the decoder generates the same codebook for reconstructing the image. If the performance of the MVQ is comparable to the MRVQ, we compress and decompress images without the inconvenience of generating and maintaining the codebook at coding and decoding ends like JPEG/DCT with advantages of VQ (fast decompression). In Figure 2, the rate distortion performance of the MVQ, MRVQ and VQ are shown. MVQ's rate distortion performance (Compression Ratio (CR) vs. Mean Squared Error (MSE)) is better than VQ. That is for a given CR, MVQ has lower distortion (MSE) compared to VQ. However, MRVQ performs marginally better than MVQ. In Figure 3, the test image and the compression results are shown. One can notice, the improvements in the visual quality of the MVQ compared to VQ and the MVQ reconstructed quality is nearly as good as the result of MRVQ. Similarly, in Figure 4, the results of VQ, MRVQ and MVQ are given for a larger compression ratio. The computational burden of computing the DCT of the codebook entries at encoding and decoding is a constant term. For large images, the constant term for decoding becomes negligibly small. Despite the fact that MRVQ scores marginally better than MVQ in rate distortion performance, we have found MVQ to be very convenient to use for compression and decompression. The elimination of the problems arising from codebooks with asymmetrical computational property makes MVQ a very useful technique in image archive and distribution application.

5. ACKNOWLEDGMENT

The authors wish to thank Kavitha Havanagi for programming support.

6. REFERENCES

1. M. Manohar and J. C. Tilton, "Model-Based VQ for image data archival, retrieval and distribution," Visual Information Processing IV, Proc of the SPIE International conference, Orlando, FL, April 17-18, 1995, pp. 190-196.

2. J. C. Tilton, M. Manohar, and J. A. Newcomer, "Earth Science Data Compression Issues and Activities,"*Remote Sensing Reviews*, Vol 9, 1994, pp. 271-298.

3. W. Pennebaker and J. Mitchell. *JPEG Still Image Data Compression Standard*, pp. 29-64, Van Nostrand Reinfold, 1993.

4. Y. Linde, A. Buzo and R. M. Gray. "An algorithm for vector quantizer design," *IEEE Trans. on Communications*, Vol. COM-28, pp. 84-95, 1980.

5. T. Kohonen. "The self-organizing map," *Proc. of the IEEE*, Vol. 78, No. 9, pp. 1464-1480, 1990.

6. P. A. Chou, T. Lookabaugh and R. M. Gray. "Optimal Pruning with Application to Tree-Structured Source Coding and Modeling," *IEEE Trans. on Information Theory*, Vol. IT-35, pp. 299-315, 1989.

7. A. Gersho and R. M. Gray. *Vector Quantization and Signal Compression*, pp. 309-406, Kluwer Academic Publishers, 1992.

8. M. Manohar and J. C. Tilton. "Progressive vector quantization on a massively parallel SIMD machine with application to multispectral image data," Accepted for publication in *IEEE Trans. on Image Processing*.

9. B. Chitprasert and K. R. Rao. "Human visual weighted progressive image transmission," *IEEE Trans. on Communications*, Vol. 38, No. 7, pp. 1040-1044, July 1990.

10. J. Ziv and A. Lempel. "A universal algorithm for sequential data compression," *IEEE Trans. on Information Theory*, Vol. IT-23, pp. 337-343, 1977.

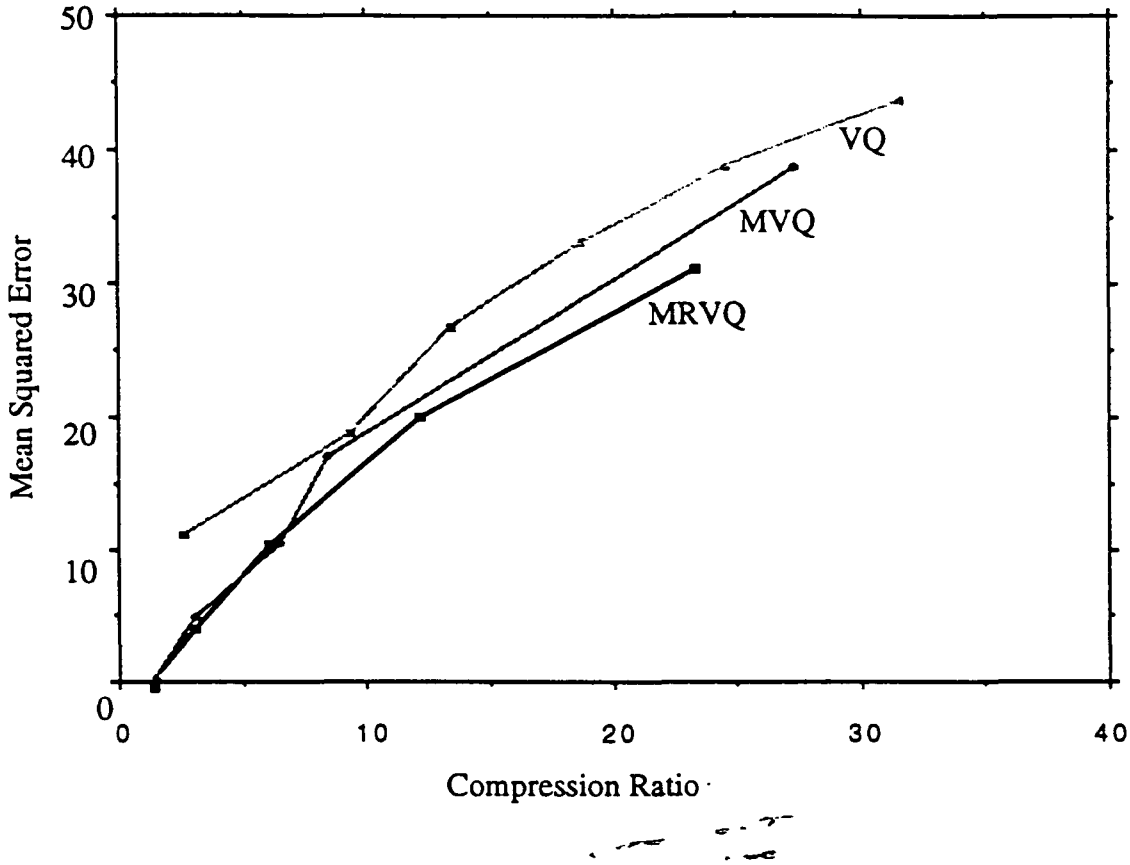

Figure 2. Rate-Distortion Performance of VQ, MVQ, and MRVQ

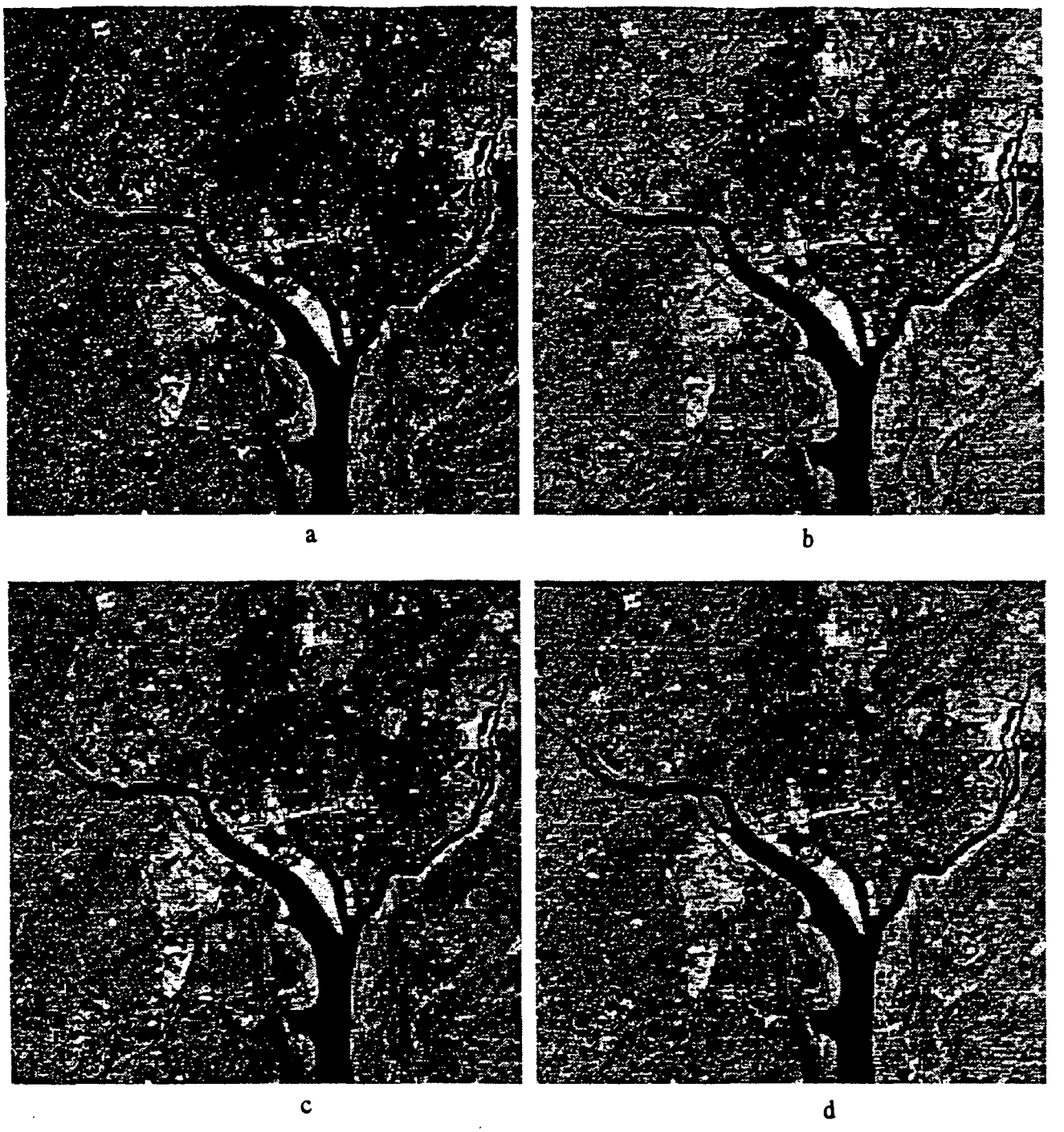

Figure 3. Visual Quality of the reconstrcuted images from VQ, MRVQ, and MVQ compression techniques. a) Landsat TM image b) VQ : rate = 0.85 bits/pixel, mse = 18.83 c) MRVQ: rate = 1.32 bits/pixel, mse = 10.42 d) MVQ: rate = 1.23 bits/pixe, mse = 10.49

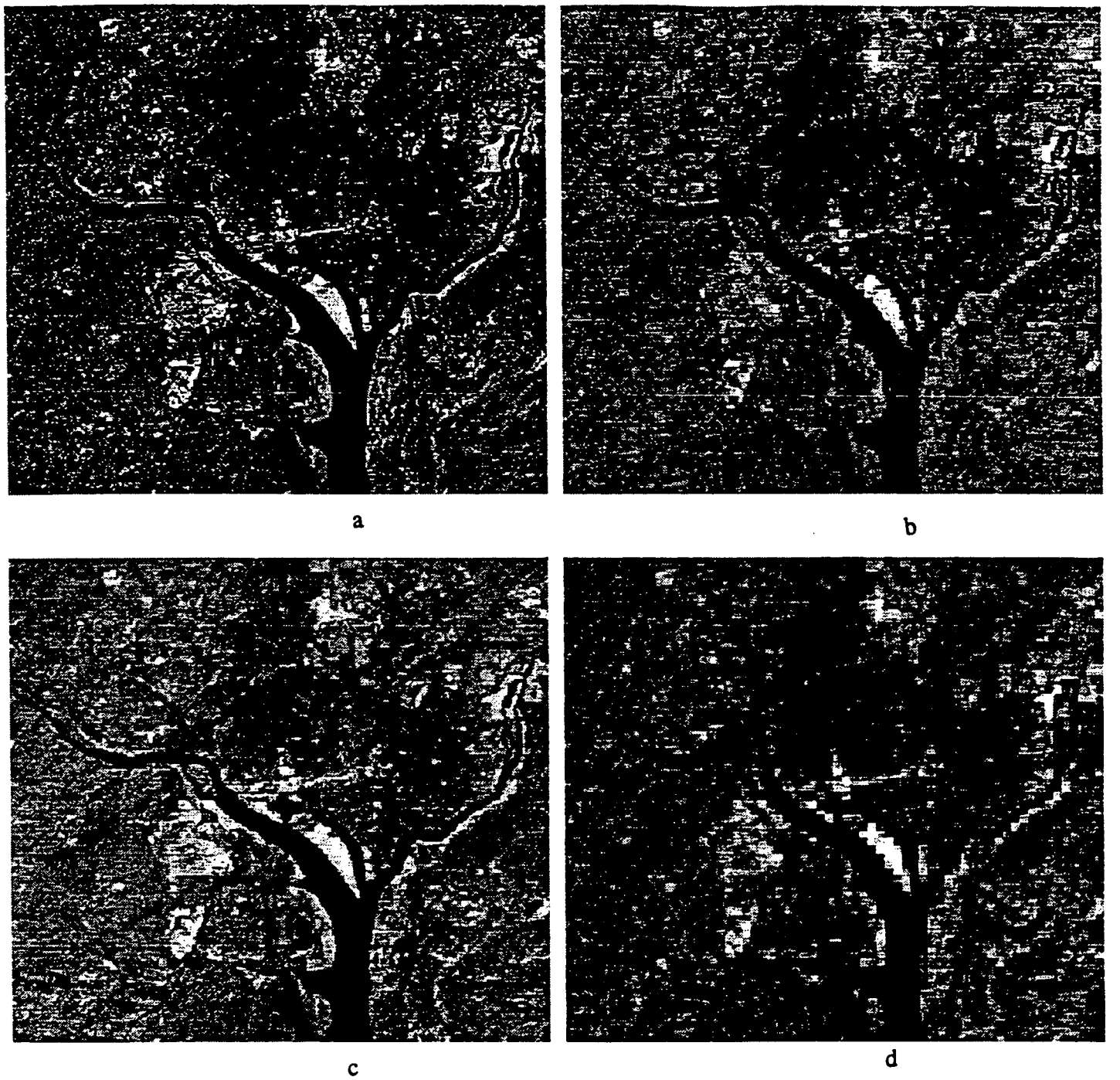

Figure 4: Visual Quality of the reconstructed images from VQ, MRVQ and MVQ at higher compression ratios. a) Landsat TM image b) VQ: rate = 0.25 bits/pixel, mse = 43.65 c) MRVQ : rate = 0.34 bits/pixel, mse = 31.18 d) MVQ rate : 0.30, mse = 38.75